

# KAN vs KAN: Examining Kolmogorov-Arnold Networks (KAN) Performance under Adversarial Attacks

Nebojsa Djosic  
Toronto Metropolitan University  
Toronto, Canada  
email: nebojsa.djosic@torontomu.ca

Evgenii Ostanin  
Toronto Metropolitan University  
Toronto, Canada  
email: eostanin@torontomu.ca

Fatima Hussain  
Toronto Metropolitan University  
Toronto, Canada  
email: fatima.hussain@torontomu.ca

Salah Sharieh  
Toronto Metropolitan University  
Toronto, Canada  
email: salah.sharieh@torontomu.ca

Alexander Ferworn  
Toronto Metropolitan University  
Toronto, Canada  
email: aferworn@torontomu.ca

**Abstract**—Recent interest in applying Kolmogorov-Arnold Networks (KANs) to the Machine Learning (ML) domain has grown significantly. Different KAN implementations leverage various architectures, with the primary distinction being their use of different learnable activation functions. While recent studies have benchmarked and evaluated the performance of different KAN models, little attention has been given to their robustness against Adversarial Attacks (AAs). In our previous work, we compared the performance of a single KAN model to a Multi-Layer Perceptron (MLP) classifier under Gaussian noise and AAs, using the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) attacks on the MNIST dataset. In this paper, we extend that analysis by comparing several popular KAN implementations subjected to the same attacks. We evaluate standard metrics, including accuracy, precision, recall, and F1-scores, using the MNIST dataset as in prior research. The aim is to empirically investigate how different activation functions influence the robustness of KAN models under AAs. Our results reveal substantial differences in accuracy loss across KAN models when exposed to AAs.

**Keywords**—FGSM; MNIST; Kolmogorov-Arnold Networks; KAN; PGD; Classification.

## I. INTRODUCTION

The fast-paced growth of Machine Learning (ML) has led to the development of increasingly advanced models that excel in various tasks. Among these innovations, Kolmogorov-Arnold Networks (KANs) introduced a novel framework grounded in the Kolmogorov-Arnold representation theorem [1]. KANs hold great potential for mobile device applications since they require less computation, memory, and thus energy to run. They also show potential for applications where interpretability is important. In addition, KAN models could be incrementally and continuously trained, although the initial training time is generally (significantly) longer for KANs, when compared to MLP models.

The increasing sophistication of adversarial attacks poses significant challenges for deep learning models, especially in safety-critical applications such as autonomous systems and cybersecurity. In addition, the growing use of ML in real-life applications are increasingly running into environmental noise

that is not present during training. Despite their potential, the robustness of KANs, especially against Adversarial Attacks (AAs) and noisy data, remains mostly underexplored in spite of several recent papers [2], [3]. Robustness is a critical aspect of machine learning models in real-world applications, which often operate under less-than-ideal conditions [4]. Adversarial attacks, such as the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) exploit weaknesses in models by introducing subtle alterations to input data, while noise can obscure important features, leading to performance degradation [5].

In our previous paper [6], we compare the robustness of one of the first KAN to that of MLP Classifier. In this paper we compare different KAN implementations each using a different learnable function. We utilize adversarial attacks such as the FGSM and PGD, which fall under the category of white-box, evasion attacks, where the attacker has full knowledge of the model and seeks to degrade performance by introducing carefully crafted perturbations to the input data. Additionally, we employ Gaussian noise as a form of non-adversarial perturbation, which can obscure critical features and simulate natural noise, further impacting model robustness. By aligning these attack methods within widely recognized taxonomies of adversarial attacks, we provide a structured approach to evaluating model vulnerabilities under both adversarial and stochastic noise conditions.

The objective of this paper is to assess the impact of these activation functions on the robustness of KAN models. The key contributions of this work include a detailed evaluation of robustness using metrics such as accuracy, precision, recall, and F1-score, a comparative analysis of model performance under various adversarial attack scenarios, and comprehensive charts and figures that visually highlight the performance differences between the models.

**Paper Structure:** The remainder of this paper is organized as follows. In Section II, we provide an overview of the related work, discussing key contributions and architecture of KAN. Section III outlines the methodology, detailing the models architecture, implementation and parameters used in

experiments. In this section we also describe the details and tools used for AAs. The experimental results are presented and analyzed in Section IV. Section V concludes the paper, summarizing the main findings, and potential directions for future work.

## II. RELATED WORK

### A. Kolmogorov-Arnold Representation Theorem

The Kolmogorov-Arnold Representation Theorem, or the superposition theorem, was introduced by Andrey Kolmogorov in 1957 and later extended and refined by Vladimir Arnold in 1963. The theorem proposes that any multivariate continuous function  $f(x_1, \dots, x_n)$  within a bounded domain can be represented as a superposition of continuous single-variable functions, which is typically written as:

$$f(x) = f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left( \sum_{p=1}^n \phi_{q,p}(x_p) \right)$$

where  $\phi_{q,p} : [0, 1] \rightarrow \mathbb{R}$  and  $\Phi_q : \mathbb{R} \rightarrow \mathbb{R}$ .

### B. KAN Architecture

KANs is a novel neural network architecture based on the Kolmogorov-Arnold representation theorem, presenting an alternative to traditional multilayer perceptrons (MLPs). Unlike MLPs, which use fixed activation functions on nodes and linear weight matrices, KANs introduced learnable activation functions along edges. Each weight in KANs is replaced by a one-dimensional learnable function, often parameterized as a spline. However, other alternatives to splines can also be used. This architectural shift promises enhanced accuracy and interpretability compared to MLPs, making KANs suitable for diverse applications in various fields [1]. An especially interesting aspect is the reduced demand for resources and the increased interpretability.

KANs and MLPs share a similar approach and architecture. Figure 1 from [1] shows KAN and MLP architectures compared side by side.

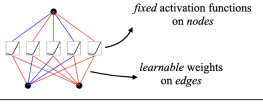
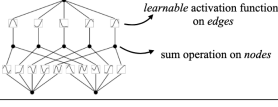
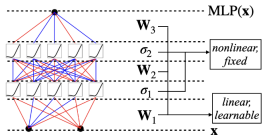
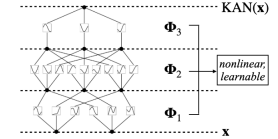
Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(x) \approx \sum_{i=1}^{N(x)} a_i \sigma(w_i \cdot x + b_i)$	$f(x) = \sum_{q=1}^{2n+1} \Phi_q \left( \sum_{p=1}^n \phi_{q,p}(x_p) \right)$
Model (Shallow)	(a) 	(b) 
Formula (Deep)	$MLP(x) = (W_3 \circ \sigma_2 \circ W_2 \circ \sigma_1 \circ W_1)(x)$	$KAN(x) = (\Phi_3 \circ \Phi_2 \circ \Phi_1)(x)$
Model (Deep)	(c) 	(d) 

Figure 1. KAN vs MLP Architectures Compared, source: [1]

Learnable activation functions along the edges are a critical component of KANs, and the choice of these functions

has a significant impact on the robustness of the model against adversarial attacks. While splines are well established for approximating the one-dimensional functions required by KAN decomposition, there are other methods capable of representing any continuous multivariate function as a finite sum of continuous univariate functions, as outlined by the Kolmogorov-Arnold theorem. Selecting the optimal one-dimensional functions is one of the most important decisions when implementing KANs, as it directly influences efficiency and model performance.

Splines are a natural choice because of their capacity to approximate continuous functions with a low number of parameters and smooth transitions between data points. This smoothness is particularly advantageous in interpolation tasks, where data changes are handled efficiently, making splines more computationally efficient compared to high-degree polynomials. Their efficiency is crucial for both the training phase and inference in KANs [1].

Beyond splines, there are several other alternative choices for learnable activation functions. Each alternative offers different trade-offs in performance and computational complexity. NNs, for instance, are capable of approximating non-linear functions with high accuracy, although training NNs for each one-dimensional function introduces longer training times. This trade-off may be beneficial for more complex tasks, where increased accuracy is needed. Polynomial function approximations are computationally straightforward and may be suitable for simpler tasks where minimal approximation is required. Fourier series offers another alternative, especially for periodic or smooth functions, using sine and cosine terms to capture the essential properties of continuous functions [7]. Chebyshev Polynomial would be another alternative to the use of splines [8].

KANs have demonstrated potential in computer vision tasks. In [9], KANs were evaluated on several well-known benchmarks, and their performance was compared to models like MLP-Mixer, Convolutional Neural Networks (CNNs), and Vision Transformers (ViTs). KANs surpassed MLP-Mixer, however, they were outperformed by the ResNet-18 model [10].

KANs sensitivity to noise is featured in [2] and [3], where authors show that even relatively small noise perturbations are causing significant degradation in KANs performance. In [11], authors show KANs weaknesses compared to MLPs in hardware applications using complex datasets requiring additional resources. Some authors like [10] claim that using KANs for more complex datasets, like CIFAR-10, shows no benefits.

### C. Adversarial Attacks

Refining techniques that exploit vulnerabilities in ML models has been the focus of recent research in AAs domain [5], [12]–[14], and particularly in computer vision [4], [15]. The FGSM and PGD emerged as the two most prominent methods for evaluating model robustness. Introduced by [16], FGSM generates adversarial examples by adding small perturbations

to input data, which can induce incorrect predictions. PGD, a more iterative and sophisticated approach developed by [17], has become a benchmark for testing resilience against stronger attacks. In computer vision, where minor changes to input images can lead to significant shifts in the model outputs, these methods are especially effective [18]. Several different defenses against FGSM and PGD have been proposed [19]–[21]. However, FGSM and PGD still remain the state of the art for assessing the robustness of ML models which is why we selected these methods to evaluate KANs in our research presented in this paper.

The Adversarial Robustness Toolbox (ART) [22] is one of many similar tools which offer techniques for generating adversarial examples and defences, while datasets like MNIST [23] are commonly used to benchmark adversarial vulnerability across studies. Although KANs have been applied in various domains, their resistance to AAs, specifically FGSM and PGD, remains largely unexamined. In this paper, we examine and compare different KAN architectures under AA attacks in an attempt to provide insights into KANs' robustness relative to the chosen architecture.

### III. METHODOLOGY

The primary objective of this study is to evaluate the relative change in performance metrics when models are subjected to adversarial attacks, rather than focusing on achieving optimal performance. While we acknowledge that each model could be fine-tuned for better results through parameter optimization and enhanced training techniques, this paper assumes that the relative impact of adversarial attacks on model performance will remain consistent. This assumption will have to be examined in future research to validate if it can be confirmed and generalized.

KAN architecture is illustrated in Figure 1. We selected four different KAN implementations available on GitHub each using a different activation function, but otherwise the same architecture as in Figure 1. All models are subjected to one noise and two AA: the FGSM, and PGD. AA are administered using ART [22]. Metrics like accuracy, precision, recall, and F1 scores are used to assess models' robustness. MNIST dataset [23], which contains 33,600 training samples and 8,400 test samples of handwritten digits, is used for all training and evaluation experiments. For a baseline, a control, we use a simple, typical MLP Classifier based on a feed forward NN. The four KAN implementations we selected to examine are: *Linear (Efficient) KAN* [24], *Naive Fourier KAN* [25], *Jacobi KAN* [26], *Chebyshev KAN* [27]

#### A. Model Architectures

We used an AdamW optimizer for all models with a learning rate of 0.001 and weight decay to prevent over-fitting. During the training Exponential Learning Rate Scheduler is used to regulate the learning rate. The CrossEntropyLoss function is used for classification.

All selected KAN models follow the same basic architecture illustrated in Figure 1 except for the activation function. The

code is provided by their respective authors on GitHub and most of the code stems from the original KAN implementation introduced in [1] and available on GitHub [28].

**Linear KAN** [24] is based on the original KAN implementation pykan [28] which is the repo behind the paper [1]. This model uses splines. It was trained using the  $(n * n) * 2 + 1$  formula derived from the Kolmogorov-Arnold theorem. The KAN NN needs to map the  $n$ -dimensional input space in this case  $n = (28 * 28)$  corresponding to the MNIST image size used as input, into one-dimensional functions, which are then summed up to recover the multivariate structure. The factor of 2 in the formula corresponds to the fact that each variable influences the other in the decomposition. Adding 1 captures the residuals, or bias, that pairwise terms may not be able to capture. The total  $784 * 784 * 2 + 1 = 1,229,377$  gives the model capacity to represent patterns in the MNIST data.

**Naive Fourier KAN** [25] replaces spline with single dimension Fourier coefficients. The authors argue that this approach would lead to a simplification since the Fourier representations are more compact and dense. The naive version uses memory proportional to the grid size parameter which is typically related to the resolution of images in our case  $28 * 28$ . Apart from the grid size this model uses bias which we set to true to allow for a flexible fit and the *smooth\_init* parameter is also set to true to help with weights initialization.

**Chebyshev KAN** [27] called *ChebyKAN* replaces spline with Chebyshev polynomials. According to the authors B-splines lead to poor performance and are not intuitive which lead to the use of the use of Chebyshev polynomials, which are widely used for approximations and polynomial interpolation since they provide close approximation to a continuous function. The simplification leads to a reduction in model parameters. Apart from the image size  $28 * 28$  and the number of classes, digits 0-9, we only have the degree of the polynomial.

**Jacobi KAN** [26] called *JacobiKAN* is based on the Chebyshev KAN [27] and it is also using orthogonal polynomials this time Jacobi. They are similar but have two extra parameters  $\alpha$  and  $\beta$  to control the upper and lower ends of the interval which is typically  $[-1, 1]$ . When these parameters are both 0 it is a special case of Jacobi, the Legendre polynomials. This is typically used for MNIST classifications. The other parameters are the same as in ChebyKAN, that is input, output, and degree.

**MLP Classifier** is a feed-forward NN in the following formation:

$$(28 * 28) \rightarrow 512 \rightarrow 256 \rightarrow 128 \rightarrow 64 \rightarrow 10$$

where  $(28 * 28)$  represents the input layer corresponding to MNIST image size. Each layer is followed by ReLU activation and dropout for regularization. The layers decrease in size leading to the output layer containing 10 neurons, one for each digit 0-9.

#### B. Attack Architecture

**Noise Attack:** We conducted Gaussian noise attacks at a noise level of 100 to evaluate the robustness of the models

under extreme conditions. This high noise level was deliberately chosen to highlight the performance degradation, enabling a clear comparison between different KAN architectures and a baseline multi-layer perceptron (MLP) model used as a control. In our prior work, we assessed the performance of a single KAN model across progressively increasing noise levels to analyze its sensitivity to noise attacks in comparison to the MLP. In this study, our focus shifts to a comparative analysis of various KAN models, while maintaining the MLP model as a reference for evaluating performance robustness.

**Fast Gradient Sign Method Attack:** ART [22] was utilized to generate adversarial examples and implement the FGSM attack on each model. Perturbations were introduced to the MNIST test data to create adversarial samples, with the epsilon parameter, typically ranging from 0.1 to 0.8, controlling the degree of perturbation. Higher epsilon values increase the likelihood of visible distortions in the images. For this study, we selected an epsilon value of 0.5, which was sufficient to degrade model performance while avoiding noticeable visual alterations to the images.

**Projected Gradient Descent Attack:** For this attack we also used the ART to prepare and run the test. The PGD attack works by progressively making small random perturbations to the input to increase (maximize) loss. In each iteration step, perturbation level is increased by a parameter while maintaining imperceptibility to the human eye through the control of the max size of perturbations. This attack is considered one of the strongest first-order adversarial attacks because of this iterative process, in contrast to the FGSM attack. We also used the 0.5 level for this attack as well, to keep it at a reasonable, realistic level.

**Tools and environment:** All KAN implementations are sourced from GitHub repositories [25]–[28] as well as The ART [22] while the MLP is implemented by us using PyTorch and Scikit-learn Python libraries. Google Colab cloud hardware and software environment is used to develop and run all experiments using Python. These standardized tools and environments along with the above listed model parameters ensure that the experiments are reproducible.

**Experiments:** All models were first trained using Google Colab free tier environments. We evaluated all models before adversarial attacks including the MLP Classifier. In each adversarial attack, we measured the change in the performance for each metric relative to the performance metrics before the attack. We also compared the performance of each KAN relative to that of MLP.

#### IV. RESULTS

**Before Attacks:** A comparison of models’ accuracy scores before attacks is visualized in the bar graph from Figure 2. It shows that MLP and Linear KAN have nearly identical accuracy. However, the other three KAN models in our experiments didn’t achieve the same level of accuracy. Since we are only interested in relative changes in metrics this was not of critical importance, however, this is something we will be looking into exploring in the future. Although not the primary focus of this

paper, we observed that KAN models took significantly more time to train. The training time in Google Colab T4 GPU free tier improved about tenfold, however, training KAN models still took more time in relatively the same proportion.

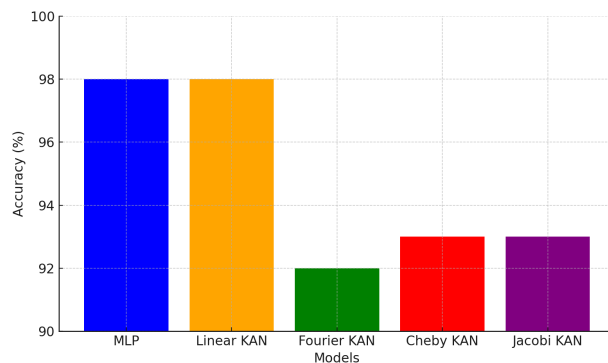


Figure 2. Model Accuracy Comparison Before Attacks

Another interesting observation is that KAN models are not as well balanced as MLP as illustrated in the graph Figure 3. Linear KAN also achieves nearly the same F1 scores across all classes except for the last, digit 9, where it significantly drops. All other KAN model F1 scores follow the same pattern. Confirming and exploring this imbalance further would be important as it may lead to interesting new directions. All these observations are left for future research.

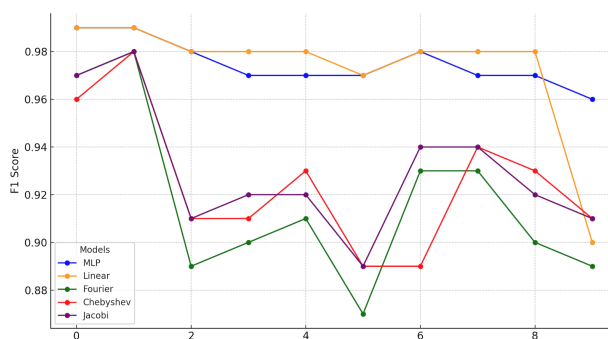


Figure 3. Model F1 Score Comparison Before Attacks

**Gaussian Noise Attack Results:** All models showed accuracy degradation when exposed to noisy data at level 100. The MLP model showed only a slight decrease followed closely by the Linear KAN model. However, the other KAN models suffered a catastrophic drop in accuracy. Figure 4 shows the before and after accuracy scores for each model. Only the Linear KAN suffered minimal accuracy degradation, although still more than double the MLP loss.

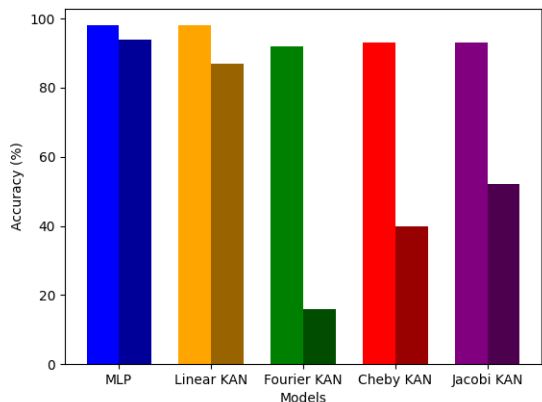


Figure 4. Model Accuracy Comparison After Noise Attack

**Fast Gradient Sign Method Attack Results:** As expected under the FGSM attacks (at 0.5 level) all models lost accuracy more than under noise attacks. Figure 5 shows the comparison between accuracy scores before and after FGSM attacks for each model. The MLP model, again, suffered the least among the models, however, this time the drop is significant. The Linear KAN model sustained even greater loss of accuracy although performed better than the rest of the KAN models. The most interesting observation following the FGSM attack is that the Fourier KAN, that suffered the worst under the noise attack, performed the best amongst KAN models, except Linear, under FGSM attack.

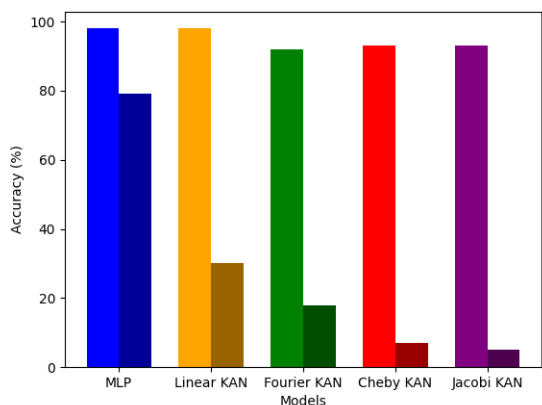


Figure 5. Model Accuracy Comparison After FGSM Attack

**Projected Gradient Descent Attack Results:** Under PGD attacks at an intensity level of 0.5, all models experienced a catastrophic degradation in performance. Figure 6 shows the comparison of accuracy scores before and after PGD attack for each model.

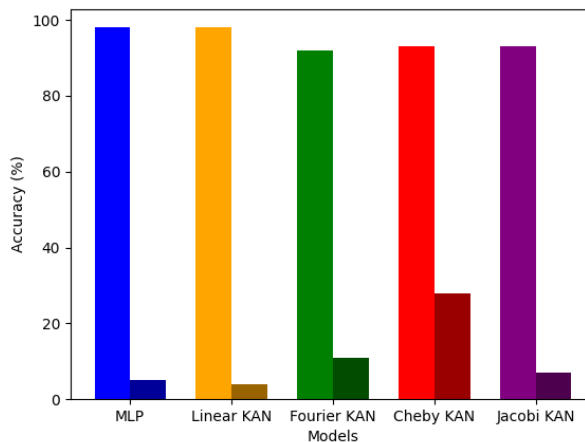


Figure 6. Model Accuracy Comparison After PGD Attack

We also observed that the accuracy for all models for the majority of digit classes dropped to zero. In the few cases where some accuracy was retained, the performance remained below 10%. What is interesting here is that all KAN models except Linear KAN, showed better overall resilience than MLP, with Cheby KAN leading the pack. Of course, the caveat is that the results are still catastrophic. The best-performing model, Cheby KAN achieved a score of just below 0.3. Still, it showed the least loss of accuracy compared to all other models.

## V. CONCLUSION AND FUTURE WORK

The results demonstrate a significant variation in how different KAN models handle AA attacks and how they compare to MLP. Consistent with our previous findings [6], we confirmed that the MLP classifier is generally more resilient than the KAN classifiers under AAs. However, under PGD attacks, we see this reversed. Specifically, the Cheby KAN model surpassed the MLP, while both the Fourier and Jacobi KAN models also achieved better performance than the MLP. We shouldn't forget that all these results are poor. Nevertheless, these preliminary empirical results highlight the need for further investigations into theoretical and empirical differences between KAN models using different activation functions, and between MLP and KAN models. Understanding these differences could offer valuable theoretical insights for both KANs and MLPs. It could potentially open new avenues for developing AA attack-resilient ML model architectures.

We also observed that KAN models have a higher class imbalance, as illustrated in Figure 3. Looking further into the reasons behind this could also unveil some interesting new insights.

In this paper, we did not prioritize training efficiency or performance optimization. However, investigating the relationship between training efficiency and robustness against AA attacks could be a valuable direction for future work, especially since KAN models offer various optimization opportunities. One notable finding is that the Cheby KAN model, despite starting with a slightly lower accuracy score of 0.93 before

attacks, compared to 0.98 for both the MLP and Linear KAN models, under PGD attacks exhibited significantly less accuracy degradation. The Cheby KAN model retained an accuracy score of 0.3, while the MLP and Linear KAN models' accuracy dropped to 0.05 and 0.04, respectively. This substantial difference warrants further exploration and could lead to important insights.

Future research directions include:

- Investigating the observed differences further and developing robustness training techniques better suited for KANs.
- Looking into training KAN models specifically aimed at handling AA.
- Looking into improving AA methods given that they were less successful attacking KAN models specifically PGD attacking the Cheby KAN model.
- Looking into the resistance of KAN models using different AA methods, and using different datasets would be a priority given our findings.

#### REFERENCES

- [1] Z. Liu *et al.*, “Kan: Kolmogorov-arnold networks”, retrieved: September 2024, Apr. 2024, [Online]. Available: <http://arxiv.org/abs/2404.19756>.
- [2] C. Zeng, J. Wang, H. Shen, and Q. Wang, “Kan versus mlp on irregular or noisy functions”, retrieved: September 2024, 2024, [Online]. Available: <https://arxiv.org/abs/2408.07906>.
- [3] H. Shen, C. Zeng, J. Wang, and Q. Wang, “Reduced effectiveness of kolmogorov-arnold networks on functions with noise”, retrieved: September 2024, Jul. 2024, [Online]. Available: <http://arxiv.org/abs/2407.14882>.
- [4] B. Xi, “Adversarial machine learning for cybersecurity and computer vision: Current developments and challenges”, *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 12, p. 1511, 5 Sep. 2020, ISSN: 19390068. DOI: 10.1002/wics.1511.
- [5] D. Dasgupta, Z. Akhtar, and S. Sen, “Machine learning in cybersecurity: A comprehensive survey”, *Journal of Defense Modeling and Simulation*, vol. 19, pp. 57–106, 1 Jan. 2022, ISSN: 1557380X. DOI: 10.1177/1548512920951275.
- [6] E. Ostanin, N. Djovic, F. Hussain, S. Shariq, and A. Ferworn, *Evaluating the robustness of kolmogorov-arnold networks against noise and adversarial attacks*, to appear in 2024 IARIA SECURWARE 2024, ECSTAI, 2024.
- [7] J. Xu *et al.*, “Fourierkan-gcf: Fourier kolmogorov-arnold network – an effective and efficient feature transformation for graph collaborative filtering”, retrieved: September 2024, 2024, arXiv: 2406.01034 [cs.LG], [Online]. Available: <https://arxiv.org/abs/2406.01034>.
- [8] S. Sidharth, A. Keerthana, R. Gokul, and K. Anas, “Chebyshev polynomial-based kolmogorov-arnold networks: An efficient architecture for nonlinear function approximation”, retrieved: September 2024, 2024, arXiv: 2405.07200 [cs.LG], [Online]. Available: <https://arxiv.org/abs/2405.07200>.
- [9] M. Cheon, “Demonstrating the efficacy of kolmogorov-arnold networks in vision tasks a preprint”, retrieved: September 2024, 2024, [Online]. Available: <https://arxiv.org/abs/2406.14916>.
- [10] B. Azam and N. Akhtar, “Suitability of kans for computer vision: A preliminary investigation”, retrieved: September 2024, Jun. 2024, [Online]. Available: <http://arxiv.org/abs/2406.09087>.
- [11] V. D. Tran *et al.*, “Exploring the limitations of kolmogorov-arnold networks in classification: Insights to software training and hardware implementation”, retrieved: September 2024, Jul. 2024, [Online]. Available: <http://arxiv.org/abs/2407.17790>.
- [12] G. Apruzzese, L. Ferretti, M. Colajanni, and M. Marchetti, “Addressing adversarial attacks against security systems based on machine learning”, in *2019 11th international conference on cyber conflict (CyCon)*, vol. 900, 2019, pp. 1–18.
- [13] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, “Adversarial machine learning”, in *Proceedings of the ACM Conference on Computer and Communications Security*, Oct. 2011, pp. 43–58. DOI: 10.1145/2046684.2046692.
- [14] K. Sadeghi, A. Banerjee, and S. K. Gupta, “A system-driven taxonomy of attacks and defenses in adversarial machine learning”, *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, pp. 450–467, 4 Aug. 2020, ISSN: 2471285X. DOI: 10.1109/TETCI.2020.2968933.
- [15] G. R. Machado, E. Silva, and R. R. Goldschmidt, “Adversarial machine learning in image classification: A survey towards the defender’s perspective”, retrieved: September 2024, Sep. 2020, [Online]. Available: <http://arxiv.org/abs/2009.03728%20http://dx.doi.org/10.1145/3485133>.
- [16] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples”, retrieved: September 2024, Mar. 2015, [Online]. Available: <https://arxiv.org/abs/1412.6572>.
- [17] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks”, retrieved: September 2024, Jun. 2017, [Online]. Available: <http://arxiv.org/abs/1706.06083>.
- [18] W. Villegas, A. Jaramillo-Alcázar, and S. Luján-Mora, “Evaluating the robustness of deep learning models against adversarial attacks: An analysis with fgsm, pgd and cw”, *Big Data and Cognitive Computing*, vol. 8, p. 8, Jan. 2024. DOI: 10.3390/bdcc8010008.
- [19] Y. Jang, T. Zhao, S. Hong, and H. Lee, “Adversarial defense via learning to generate diverse attacks”, in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2740–2749.
- [20] G. Sriramanan, S. Addepalli, A. Baburaj, and R. V. Babu, “Guided adversarial attack for evaluating and enhancing adversarial defenses”, in *Advances in Neural Information Processing Systems*, retrieved: September 2024, vol. 33, 2020, pp. 20297–20308.
- [21] S. Mohandas, N. Manwani, and D. P. Dhulipudi, “Momentum iterative gradient sign method outperforms pgd attacks”, in *International Conference on Agents and Artificial Intelligence*, vol. 3, Science and Technology Publications, Lda, 2022, pp. 913–916. DOI: 10.5220/0010938400003116.
- [22] M.-I. Nicolae *et al.*, “Adversarial robustness toolbox v1.0.0”, retrieved: September 2024, 2019, [Online]. Available: <https://arxiv.org/abs/1807.01069>.
- [23] L. Deng, “The mnist database of handwritten digit images for machine learning research”, *IEEE Signal Processing Magazine*, vol. 29, pp. 141–142, 6 2012, ISSN: 10535888. DOI: 10.1109/MSP.2012.2211477.
- [24] H. Cao, “An efficient implementation of kolmogorov-arnold network (kan)”, retrieved: September 2024, 2024, [Online]. Available: <https://github.com/Blealtan/efficient-kan>.
- [25] G. Noesis, “Pytorch layer for fourierkan”, retrieved: September 2024, 2024, [Online]. Available: <https://github.com/GistNoesis/FourierKAN/tree/main>.
- [26] SpaceLearner, “Jacobi polynomials kan”, retrieved: September 2024, 2024, [Online]. Available: <https://github.com/SpaceLearner/JacobiKAN>.
- [27] SynodicMonth, “Chebyshev polynomials kan”, retrieved: September 2024, 2024, [Online]. Available: <https://github.com/SynodicMonth/ChebyKAN/>.
- [28] Z. Liu, “Python kolmogorov-arnold networks (kans)”, retrieved: September 2024, 2024, [Online]. Available: <https://github.com/KindXiaoming/pykan>.